

Table based AHC Variants for Text Clustering

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based AHC variants as the approaches to the text clustering. The initial AHC algorithm was modified into the table-based version as an approach to the task. In this research, in the view of the numerical vector-based versions, we mention the three AHC variants, one where the data clustering proceeds in the bottom-up direction with the similarity threshold, one where it allows the merge of more than two pairs, and one where clusters are merged based on the radius. The three AHC variants are modified into the table-based versions as the text clustering tools, as well as the initial AHC algorithm. The goal of this research is to improve the clustering performance by proposing the modified versions of the AHC variants.

I. INTRODUCTION

The text clustering is defined as the process of partitioning a single text group into several subgroups based on text contents. An entire group of texts is given as the input, and each subgroup in the clustering results which are generated as the output contains content based similar texts. The process of text clustering is to encode texts into structured data, define the similarity metric between structured ones as the text representations, and apply unsupervised learning algorithm to the task. The task of this research is the hard text clustering where each text is arranged into only one cluster, and it will be expanded into the soft text clustering or the hierarchical text clustering in subsequent research. The text clustering is viewed as the process of automating the preliminary tasks for the text categorization such as the category predefinition and the sample text allocation.

This research is motivated by the successful results from applying the table based AHC algorithm to the text clustering in the previous work. Encoding texts into tables and defining the similarity metric between tables was the solution to the main problems in encoding texts into numerical vectors. The AHC algorithm was modified into the table-based version as the approach to the text clustering by the similarity metric between tables. The modified AHC version which process tables directly was empirically validated in the text clustering as its better performance than the traditional version. The differences of this research from the previous work are to modify the AHC variants into the table-based versions and to apply them to the text clustering.

Let us mention the differences of the AHC variants from the AHC algorithm which was modified into the table-

based version in the previous work. In the first variant, the similarity threshold is used as a parameter in proceeding the data clustering in the bottom-up direction. In the second variant, it is possible to merge more than one pair of clusters in each iteration. In the third variant, the radius from a particular cluster is the criteria for merging clustering clusters in each iteration. In this research, we propose that that three AHC variants should be modified into the table-based versions as the approaches to the text clustering.

Let us mention some benefits from this research. Encoding texts into tables is the solution to the main problems in encoding texts into numerical vectors. The clustering speed is improved by replacing the AHC algorithm by the AHC variants. The clustering performance is improved by modifying the AHC variants into the table-based versions. More recommendable approaches are provided for implementing the text clustering system by improving both the clustering speed and the clustering performance.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the AHC algorithm to its three variants. The directions of exploring previous works are derivation of AHC variants, modification of the standard AHC algorithm into its table-based version, and the cases of encoding a text into a table. The distinguishable points of this research are derivation of three AHC variants from the standard version, modification of them into table-based versions, and their applications to the text clustering. This article is intended to explore previous works for providing the background for this research.

Let us explore the previous cases of deriving AHC variants. In 2012, Murtagh and Contreras mentioned the possibility of deriving various versions of AHC algorithm and a particular variant which uses nearest neighbors [1]. In 2013, Sasirekha and Baby presented various similarity

metrics between vectors and strategies of defining a similarity between clusters [2]. In 2015, Nazari et al. proposed the AHC algorithm which merges two clusters based on their common nearest neighbors [6]. In the AHC algorithms which are mentioned in above previous literatures, only one pair of clusters is merged in each iteration.

Let us explore the previous works which are concerned with the application of the modified version of the standard AHC algorithm to text clustering. In 2017, Jo initially proposed that the modified version should be applied to the text clustering [7]. In 2019, Jo validated empirically the table-based AHC algorithm in the text clustering [8]. In 2023, he prepares the journal article which describes the table-based AHC algorithm and its application to the text clustering for its publication [9]. What is provided by the previous works becomes the basis for this research.

Let us explore the previous works on the table matching algorithm as the early case of encoding a text into a table. In 2008, the table based matching was proposed as an approach to the text classification as the initial case of encoding a text into a table, instead of a numerical vector [3]. In 2011, the table based matching algorithm was registered as a patent by Jo [4]. In 2015, it was upgraded into its more reliable and stable version by Jo [5]. The similarity metric between two tables is provided by the above previous works for modifying the standard KNN algorithm and its variants.

Let us mention the differences of this research from the previous works which are explored above. In this research, we derive the AHC variants as fast clustering algorithms by allowing merging multiple cluster pairs for each iteration. The three AHC variants are modified into table-based versions and adopted for implementing the text clustering system. In this research, the table based AHC variants are developed as clustering algorithms as well as classification algorithms. We will describe the modified AHC variants in Section III.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a text into a table and the proposed similarity metric. In Section III-B, we describe the standard version of AHC algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three variants from the standard version. In Section III-D, we present the system architecture of the text clustering system.

A. Encoding Process

This section is concerned with the table as the text representation. It is defined as a set of entries, each of which consists of an element and its weight. The table which represents a text consists of entries, each of which is a pair of a word and its weight in the text. The similarity between tables is computed based on shared words as the semantic

similarity between texts. This section is intended to describe the process of encoding a text into a table and the process of computing the similarity between tables.

The process of encoding a text into a table is illustrated in Figure 1. It is indexed into a list of words. The weights of words in the text are computed in the text-word weighting; the word weight in the text is a relationship between a word and a text. The entries with their lower weights are removed for downsizing the table. Refer to [3] for studying the entire process in more detail.

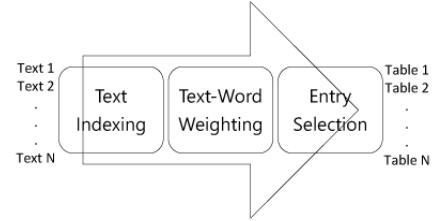


Figure 1. Process of Encoding Texts into Tables

Let us mention the function of a table which maps it into a set of words as shown in Figure 2. The table which represents a text is expressed as entry set as shown in equation (1),

$$T = \{(t_1, w_1), (t_2, w_2), \dots, (t_{|T|}, w_{|T|})\} \quad (1)$$

where t_i is a word, and w_i is the weight of the word, t_i , in the text. The function for generating a list of words is expressed as equation (2),

$$F(T) = \{t_1, t_2, \dots, t_{|T|}\} \quad (2)$$

The function is the operation which takes only words without their weights as a set. The operation is applied to computing the similarity between tables which represent texts.

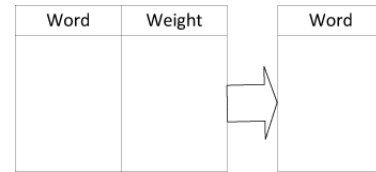


Figure 2. Conversion of Table into Word Set

Let us mention the process of computing the similarity between two tables as the similarity between two texts. The tables which represent texts are notated by the two sets: $D_1 = \{(t_{11}, w_{11}), (t_{12}, w_{12}), \dots, (t_{|D_1|}, w_{|D_1|})\}$ and $D_2 = \{(t_{21}, w_{21}), (t_{22}, w_{22}), \dots, (t_{|D_2|}, w_{|D_2|})\}$. The intersection between the two sets which are generated by applying the function, $F(\cdot)$, by equation (3),

$$F(D_1) \cap F(D_2) = \{t_{s1}, t_{s2}, \dots, t_{s|F(D_1) \cap F(D_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared word, its weight from the table, D_1 , its weight from the

table, D_2 , is constructed based on equation (3), as expressed in equation (4),

$$SD_{12} = \{(t_{s1}, w_{s1}^1, w_{s1}^2), (t_{s2}, w_{s2}^1, w_{s2}^2), \dots, (t_{s|F(D_1) \cap F(D_2)|}, w_{s|F(D_1) \cap F(D_2)|}^1, w_{s|F(D_1) \cap F(D_2)|}^2)\} \quad (4)$$

The similarity between the two tables, D_1 and D_2 , is computed by equation (5),

$$sim(D_1, D_2) = \frac{2(\sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^1 + \sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^2)}{\sum_{i=1}^{|D_1|} w_{1i} + \sum_{i=1}^{|D_2|} w_{2i}}. \quad (5)$$

The value which is computed by equation (5), is always given as a normalized value between zero and one, as shown in equation (6),

$$0 \leq sim(D_1, D_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a table by the text indexing, the word weighting, and the entry selection. A table is expressed as a set of entries, and a table is filtered into a set of words by the defined function. The similarity between tables is computed based on shared words by equation (5). In the future research, we consider representing a text into multiple tables.

B. Initial Version of Table based AHC Algorithm

This section is concerned with the initial version of table based AHC algorithm. The version of AHC algorithm was initially proposed as the approach to the text clustering by Jo in 2019 [8]. The similarity metric between tables which was described in the previous section is used for computing the similarity between clusters. The AHC algorithm proceeds clustering data items with the bottom-up direction. This section is intended to describe the initial version of AHC algorithm from which its variants are derived.

The initial status for applying the AHC algorithm to the word clustering is illustrated in Figure 3. The words as clustering targets are encoded into tables by the process, which was described in Section III-A, and they are notated by a set, $Tr = \{T_1, T_2, \dots, T_N\}$. The clusters are defined as many as data items, as $C_1, C_2, \dots, C_N$, and each cluster includes a data item as $C_i = \{T_i\}$. The initial status which is represented in Figure 00 is notated by $C_1 = \{T_1\}, C_2 = \{T_2\}, \dots, C_N = \{T_N\}$. The cluster with only one item is called singleton, and singletons are constructed as many as data items in the initial status.

The process of computing the similarity between two clusters is illustrated in Figure 4. The two clusters are notated by two sets of items, $C_1 = \{T_{11}, T_{12}, \dots, T_{1|C_1|}\}$



Figure 3. Initial Clusters in using AHC Algorithm: Singletons

and $C_2 = \{T_{21}, T_{22}, \dots, T_{2|C_2|}\}$. The similarity between clusters, C_1 and C_2 , is computed by equation (7),

$$sim(C_1, C_2) = \frac{1}{|C_1| \cdot |C_2|} \sum_{i=1}^{|C_1|} \sum_{j=1}^{|C_2|} sim(T_{1i}, T_{2j}). \quad (7)$$

If the similarity metric which was described in Section III-A is adopted, the similarity between two clusters is always a normalized value between zero and one, as expressed in equation (8),

$$0 \leq sim(C_1, C_2) \leq 1. \quad (8)$$

The complexity of computing the similarity between two clusters is expressed by $O(|C_1||C_2|)$.

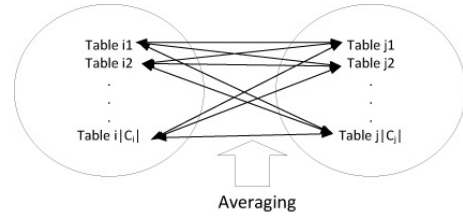


Figure 4. Similarity between Clusters

The process of clustering data items is illustrated as pseudo codes in Figure 5. The AHC algorithm begins with the status where singletons are given as many as data items. All possible pairs of clusters are generated from the current cluster list, their similarities are computed by equation (7), and the cluster pair with the highest similarity is merged into a cluster. In the AHC algorithm, the data items are clustered by iterating computing similarities of all possible cluster pairs and merge a cluster pair into a cluster. In each iteration, one cluster is decreased, and this iteration continues until reaching the desirable number of clusters.

Let us make some remarks on the initial version of the AHC algorithm from which we derive the variants. The initial status in applying the AHC algorithm to data clustering is that there are singletons as many as data items. The similarity between clusters is computed by averaging the similarities of all possible pairs from them. Data items are clustered by iterating computing the similarities of all possible cluster pairs and merge the pair with its maximal similarity into one cluster. In its variant, we will consider allow more than one pair of clusters to be merged.

C. AHC Variants

This section is concerned with the variants which are derived from the AHC algorithm which was covered in

```

List clusterDataItem(List tableList, int desiredClusterSize){
    int itemSize = tableList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Table dataItem = tableList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterList.size() > desiredClusterSize){
        double maxSimilarity = 0.0;
        int maxIndex1 = 0;
        int maxIndex2 = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1,c2);
                if(similarity > maxSimilarity){
                    maxSimilarity = similarity;
                    maxIndex1 = i;
                    maxIndex2 = j;
                }
            }
        }
        Cluster cmax1 = clusterList.getElement(maxIndex1);
        Cluster cmax2 = clusterList.getElement(maxIndex2);
        Cluster mergeCluster = cmax1.merge(cmax2);
        clusterList.deleteElement(cmax1);
        clusterList.deleteElement(cmax2);
        clusterList.addElement(mergeCluster);
    }
}

```

Figure 5. Initial Version of AHC Algorithm

Section III-B. The difference from the traditional version of AHC algorithm is to adopt the similarity metric between tables for computing the similarity between clusters. In this section, we derive the three variants by merging cluster pairs by adopt the threshold-based selection, allowing merge of multiple pairs in each iteration, and merging clusters within the radius. In this research, we adopt the three variants of AHC algorithm for implementing the word clustering system. This section is intended to describe the three variants of AHC algorithm.

In Figure 6, we illustrate the process of clustering data items by the first variant of AHC algorithm. The clusters are initialized with singletons like the initial version which was described in Section III-B. All possible cluster pairs are generated, and the cluster pairs whose similarities are more than or equality to the similarity threshold are merged. If there is no pair which satisfies the condition, clustering data items is terminated. In this variant, the number of clusters is decreased variably in this variant.

We illustrate the process of clustering data items by the second variant of AHC algorithm in Figure 7 as a pseudo code. The number of cluster pairs which are merged into each iteration is given as an additional parameter in this variant. All possible cluster pairs are generated, and the similarity is computed for each pair. The cluster pairs are sorted by the descending order of the similarity, and the pairs within the rank are merged. The difference of this variant from the initial version is to merge multiple cluster pairs,

```

List clusterDataItem(List tableList, double similarityThreshold){
    int itemSize = tableList.size();

    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Table dataItem = tableList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    boolean similarityFlag = false;
    //Cluster Data Items in the bottom up direction
    while(similarityFlag){
        int clusterListSize = clusterList.size();
        similarityFlag = false;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1,c2);
                if(similarity >= similarityThreshold){
                    Cluster mergeCluster = c1.merge(cmax2);
                    clusterList.updateElement(i,mergeCluster);
                    clusterList.deleteElementIndex(j);
                    clusterListSize = clusterList.size();
                    similarityFlag = true;
                }
            }
        }
    }
}

```

Figure 6. Feature Similarity based AHC Variant 1: Similarity Threshold Based AHC Algorithm

instead of one pair.

```

List clusterDataItem(List tableList, double similarityThreshold){
    int itemSize = tableList.size();

    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Table dataItem = tableList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterListSize > desiredClusterSize){
        List pairList = new List();
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = i; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                Pair indivPair = new Pair(c1,c2);
                indivPair.computeSimilarity();
                pairList.addElement(indivPair);
            }
        }
        pairList.sortPairs();
        List selectedPairList = pairList.getSubList(0,rank-1);
        clusterList.mergeClusters(selectedPairList);
        clusterListSize = clusterList.size();
    }
}

```

Figure 7. Feature Similarity based AHC Variant 2: Merge of Multiple Pairs

In Figure 8, we illustrate the process of clustering data items by the last variant of AHC algorithm. The similarity threshold is given as an external parameter, and the number of other clusters whose similarity is greater than or equality to the similarity threshold is counted. In each iteration, the cluster with its maximal number of its similar clusters is appointed as the pivot, and the pivot cluster and its similar ones are merged into one cluster. This variant iterates

selecting the pivot cluster in the current cluster list and merge the pivot and its similar ones, as the process of clustering data items. If there is no cluster with its similar cluster, the iteration is terminated.

```

List clusterDataItem(List tableList, double similarityThreshold){
    int itemSize = tableList.size();

    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        Table dataItem = tableList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    //Cluster Data Items in the bottom up direction
    while(true){
        int clusterListSize = clusterList.size();
        if(clusterListSize == 1)
            return;
        int maxSimilarClusterSize = 0;
        int maxIndex = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            int similarClusterSize = c1.countSimilarClusters(clusterList);
            if(similarClusterSize > maxSimilarClusterSize){
                maxSimilarClusterSize = similarClusterSize;
                maxIndex = i;
            }
        }
        if(maxSimilarClusterSize == 0)
            return;
        Cluster pivotCluster = clusterList.getElement(maxIndex);
        List similarClusterList = clusterList.getSimilarClusterList(pivotCluster);
        clusterList.mergeClusterList(similarClusterList);
    }
}

```

Figure 8. Feature Similarity based AHC Variant 3: Radius based AHC Algorithm

Let us make some remarks on the three variants of AHC algorithm which are adopted as the approaches to the word clustering. In each iteration of the first variant, cluster pairs with their similarities which are greater than or equal to the threshold are merged. In each iteration of the second variant, a fixed number of cluster pairs with their highest similarities are merged. In each iteration of the third variant, a pivot cluster and its similar clusters are merged. We may derive more variants from the AHC algorithm by setting different policies on merging clusters.

D. System Architecture

This section is concerned with the architecture and the execution flow of the text clustering system. In Section III-C, we describe the table based AHC variants and adopt them for implementing the text clustering system. In the architecture of the text clustering system, which was previous proposed, the initial AHC algorithm which is mentioned in Section III-B is replaced by its variants. The process of executing the system is to encode the texts into tables and cluster them. This section is intended to describe the text clustering system which is implemented in this study with respect to its architecture.

A group texts and numerical vectors is illustrated in Figure 9. All of texts are initially given at a time under the assumption of the offline clustering. The texts in the group

are encoded into tables by the process which is described in Section III-A. They are clustered by the AHC algorithm which is described in Section III-C. The online clustering where texts are given as a stream will be considered in next research.

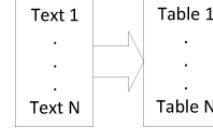


Figure 9. Preparation of Training Examples

The architecture of the text clustering system is illustrated in Figure 10. In the encoding module, texts as clustering targets are encoded into tables. The clustering module includes the similarity computation module which computes the similarities among the tables by the process which is described in Section III-A and clusters them by one of the AHC variants. The text clusters are generated as the final output from the system. The difference from the system architecture which was proposed in [8] is to replace the initial AHC algorithm by its variants.

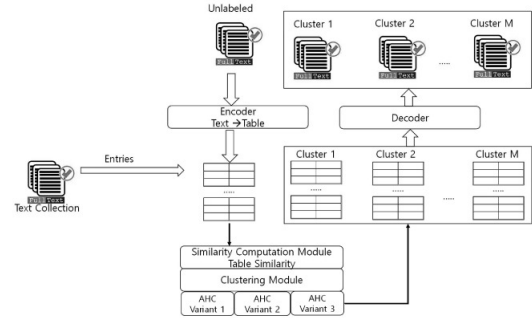


Figure 10. System Architecture

The execution flow of the text clustering system is illustrated in Figure 11. The texts are encoded into tables. The similarity metric between clusters which is expanded from one between tables is used in the AHC variants. The data items are clustered, selecting one among the three AHC variants which are described in Section III-C. The external parameter, the similarity threshold, is controlled in the first and the third variant for reaching the desired number of clusters.

Let us make some remarks on the proposed version of the text clustering system whose architecture is presented in Figure 10. The offline clustering is assumed in implementing the system; it will be expanded for doing online clustering in the next research. The upgrading point of the system is to replace the initial AHC algorithm by its variants which are described in Section III-C. The texts are encoded into tables, the similarity between clusters is computed by equation (7), and they are clustered by one of the AHC variants. It

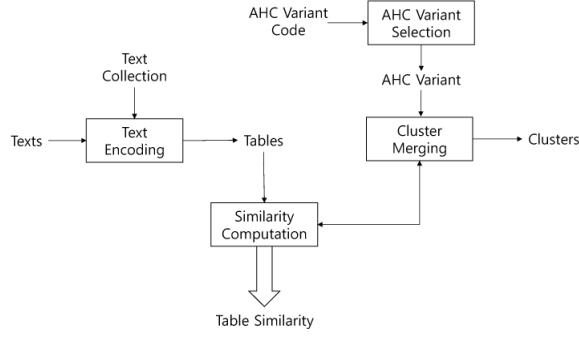


Figure 11. System Flow

is expected that the clustering performance and speed are improved, by replacing the initial version by its variants.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We derive the three variants from the standard version by allowing multiple clusters to be merged at a time. We encode words into tables and apply the similarity metric among them to the AHC variants as well as the standard version. We design the text clustering system by adopt the AHC variants with the proposed similarity metric. In the next research, we will implement the text clustering system as a real system.

Let us mention the remaining tasks as the further discussions on this research. We consider representing a text into multiple tables and define the similarity between two table sets. We may modify other machine learning algorithms, such as the Naïve Bayes and the SVM (Support Vector Machine), into the table-based versions. We may apply the proposed versions of the AHC variants to other clustering tasks. The text clustering may be implemented by adopting the proposed AHC variants as real programs.

REFERENCES

- [1] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview", 86-97, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2, 1, 2012.
- [2] K. Sasirekha and P. Baby, "Agglomerative hierarchical clustering algorithm-A Review", 83-85 International Journal of Scientific and Research Publications, 3,1, 2013.
- [3] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, 11, 2008.
- [4] T. Jo, "Device and Method for Categorizing Electronic Document Automatically", 10-2009-0041272, 10-1071495, 2011.
- [5] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, 19, 4, 2015.
- [6] Z. Nazari, D. Kang, M.R. Asharif, Y. Sung, and S. Ogawa, "A new hierarchical clustering algorithm", 148-152, The Proceedings of IEEE International Conference on Intelligent Informatics and Biomedical Sciences, 2015.
- [7] T. Jo, "Table based AHC for Text Clustering", 133-138, The Proceedings of 13th International Conference on Data Mining, 2017.
- [8] T. Jo, "Applying Table based AHC Algorithm to News Article Clustering", 8-11, The Proceedings of International Conference on Green and Human Information Technology, Part I, 2019.
- [9] T. Jo, "Similarity Metric between Tables for Modifying AHC Algorithm in Text Clustering", in Preparation, 2023.